BOLT BERANEK AND NEWMAN INC

CONSULTING · DEVELOPMENT · RESEARCH

BBN Report No. 3257

February 1976

DISTRIBUTED COMPUTATION AND TENEX-RELATED ACTIVITIES

Quarterly Progress Report No. 5

1 November 1975 to 30 January 1976

This research was supported by the Defense Advanced Research Projects Agency under ARPA Order No. 2901. Contract No. N00014-75-C-0773.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

(14) REPORT NUMBER: BBN [redacted]-3257

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

(6) TITLE (and Subtitle): DISTRIBUTED COMPUTATION AND TENEX-RELATED ACTIVITIES,

(9) 5. TYPE OF REPORT & PERIOD COVERED: Quarterly Progress Rept. 1 Nov - 30 Jan 76

6. PERFORMING ORG. REPORT NUMBER

(10) 7. AUTHOR(s): J. Burchfiel, R. Thomas, T. Myer, R. Tomlinson

(15) 8. CONTRACT OR GRANT NUMBER(s): N00014-75-C-0773, ARPA Order-2935

9. PERFORMING ORGANIZATION NAME AND ADDRESS:
Bolt Beranek and Newman Inc.
50 Moulton Street
Cambridge, Massachusetts 02138

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

(12) 29 p.

11. CONTROLLING OFFICE NAME AND ADDRESS

(11) 12. REPORT DATE: Feb[redacted]76

13. NUMBER OF PAGES: 25

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

15. SECURITY CLASS. (of this report): UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution of this document is unlimited. It may be released to the Clearinghouse, Department of Commerce for sale to the general public.

(16) ARPA Order-2935

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

This research was supported by the Defense Advanced Research Projects Agency under ARPA Order No. 2935.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

distributed computation
TENEX security
message processing
National Software Works
CINCPAC Interactive Test

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes BBN efforts in the design of the National Software Works system and BBN efforts to integrate TENEX into the National Software Works System. It also describes BBN efforts to upgrade existing ARPANET message service to meet Navy requirements for an interactive message processing test at CINCPAC.

FORM DD 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

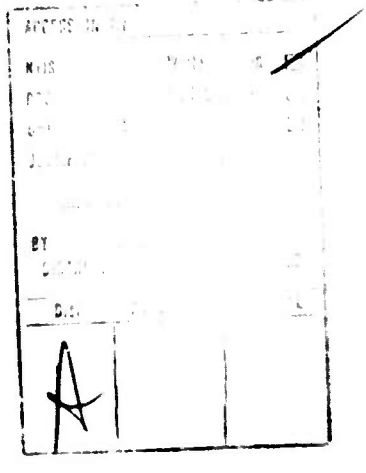
UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

060 100

15

TABLE OF CONTENTS



I. Introduction

This report describes our ARPA research in "Distributed Computation and TENEX Related Activities" for the period November 1975 through January 1976. During this period our research continued in the areas of:

- The National Software Works
- TENEX enhancements
- Message Technology

Details of this work are included in the sections of this report that follow.

Work on the National Software Works (NSW) project was concentrated in three areas: preparations for the initial demonstration of the NSW system; design of the NSW interprocess communication facility; and design of the NSW Foreman component. The demonstration of NSW held at Gunter in November was quite successful. We have completed the design of the NSW interprocess communication facility and we have started to implement the facility for the TENEX family of hosts. We expect the Foreman specification to be completed during the current quarter. Implementation of an NSW Foreman for the TENEX hosts should begin toward the end of the current quarter

Our major efforts in the area of message technology have been to move ahead with the development and refinement of MAILSYS/HERMES Version 2. Our work here has followed the phased development plan described in our last report. To a large extent, our current efforts are being guided by the human factors analysis of the early, experimental states of HERMES 2. In a related effort our human factors group is developing documentation to provide tutorial support for users of the system with varying levels of sophistication in computer technology. Our work on the LDMX/TENEX interface has progressed to the point where the design of the TENEX portion of the protocols is complete and implementation is well underway.

II. Distributed Computation

The National Software Works (NSW) project is presently the primary focus of our work in distributed computation. During the period of November through January our work on the NSW was concentrated in three areas: final preparations for the demonstration of the NSW system at Gunter; design of the NSW interprocess communication facility; and, design of the NSW Foreman component. Throughout this period we have continued to work closely with Massachusetts Computer Associates on the NSW project. As part of the work cited above we participated in a number of meetings with other NSW participants during this quarter. In addition to the November 17-19 Gunter demonstration/meeting, there were numerous other meetings between NSW participants.

On 15 December 1975 BBN hosted groups from Computer Associates and MIT to discuss the communications needs of NSW and to present BBN proposals in these areas. As an outgrowth of agreements reached at this meeting, the developing concepts were presented to a wider NSW group in a meeting held at the Compass offices December 15-16. On 15 January 1976 we convened a follow-up meeting of the NSW protocol committee to finalize the initial MSG design specification. During January we also hosted a group from the Naval Electronic Laboratory Center (NELC). This group is to be an initial NSW user group, and we briefed them on pertinent NSW and TENEX issues. A series of meetings began in January with Works Manager (WM) personnel to discuss aspects of the Foreman/WM interaction.

A. Preparations for NSW Demonstration.

Early in this quarter we completed initial versions of our TENEX TBH software, in time to be initially demonstrated as part of an overall NSW demonstration at Gunter AFB, Alabama. This included a TENEX encapsulator program suitable for handling the text editors TECO and SOS, and a file package capable of performing local file operations. Both components were fully integrated with the other NSW components using the interim intra-TENEX communication facility, to help demonstrate a primitive TENEX-IBM360 NSW capability.

In order to have this initial capability more accurately reflect the nature of the NSW effort, we completed the implementation of an ARPANET server module which invoked the automatic startup of the NSW. (The design considerations for such a module were described in the previous QPR). Utilizing this network server means that the NSW user need not log into TENEX to run the TENEX based NSW system. He is automatically connected to the TENEX CLI process for logging into the NSW system directly.

The NSW server essentially creates a new TENEX job in response to connection requests to the NSW socket. The job is logged in as the generic NSW user, and the execution of the NSW software is automatically started. The WM ensures proper login to NSW, and the user cannot access the TENEX EXEC. Thus he sees just the TENEX based NSW system. The NSW job dispatcher will continue to be useable, as the NSW system itself expands to a multi-computer system.

The Gunter demonstration was deemed to be very successful, with the TENEX TBH components performing their tasks as specified. The file package, encapsulator and dispatcher form a solid basis on which we can build toward more capable network oriented TENEX TBH software.

B. MSG - The Interprocess Communication Facility for NSW.

During this quarter, together with Massachusetts Computer Associates, we designed a facility to support the interprocess communication requirements of the NSW system. This communication facility (named MSG) is designed to support communication among processes distributed over a number of computer systems connected by a computer network such as the ARPANET.

We began the MSG design by identifying the important processes which comprise NSW and the patterns of communication these processes require. Next we abstracted from those patterns a model of interprocess communication which was sufficient for the NSW. Finally we developed the details of the MSG facility itself.

The details MSG, as well as the analysis leading to its design, is documented in the paper "MSG: The Interprocess Communication Facility for the National Software Works" [BBN Report 3237, Compass Report CADD-7601-2611]. As part of our design effort, we also prepared a series of notes (BBN NSW Working Notes Numbers 1 through 11) detailing our thoughts and positions on various NSW communication issues. These notes are available on request.

We have just started to implement the MSG facility for the TENEX hosts. We expect that this initial implementation will be substantially complete by the end of the next quarter.

The following is a brief overview of the MSG interprocess communication facility; this overview was abstracted from BBN Report 3237.

MSG supports the NSW patterns of communication by providing two different modes of process addressing:

- generic addressing;
- specific addressing;

and three different modes of communication:

- messages;
- direct communication paths (connections);
- alarms.

Each mode of process addressing and communication is intended to satisfy certain NSW requirements and to be used in certain kinds of situations. However, MSG itself does not impose any limitations on how processes use the various communication modes. MSG does not interpret messages or alarms, nor does it intervene in communication on direct connections. The interpretation of messages, alarms, or direct connections is entirely a matter for the processes using MSG to communicate.

Generic addressing is used by processes which either have not communicated before or for which the details of any past communication is irrelevant. It is restricted to the message mode of communication. A valid generic address specifies a functional process class. When MSG accepts a generically addressed message it selects as destination some process which is not only in the generic class addressed but has also declared its willingness to receive a

generically addressed message. If there is no such process, MSG may create one.

A valid specific address refers to exactly one process and this address remains valid for the life of that process. Specific addressing may be used with all three communication modes. Specific addressing is used between processes which are familiar with each other. The familiarity is generally because the processes have communicated with each other before, either directly or through intermediary processes.

Message exchange is expected to be the most common mode of communication among NSW processes. To send a message, a process addresses it by specifying the address of the process to receive the message and then executes an MSG "send" primitive which requests MSG to deliver the message. When MSG delivers a message to a process it also delivers the name (i.e., specific address) of the process that sent the message.

The second mode of MSG communication is direct access communication. A pair of processes can request that MSG establish a direct communication path between them. Direct communication paths are provided to support certain NSW communication requirements such as file transfers between hosts and terminal-like communication between a Front End and tool/Foreman. (The ARPANET realization for a direct communication path is a host/host connection or connection pair.)

The alarm mode of communication is supported by MSG to satisfy a communication requirement typically satisfied by interrupts in other interprocess communication systems. Alarms provide a means for one process to alert another process to the occurrence of an exceptional or unusual event. Processes may send and receive alarms much as they send and receive messages. However, there are significant differences between alarms and messages. The rules that govern the flow and delivery of alarms are different from those that govern the flow and delivery of messages. In particular, the delivery of an alarm to a process is independent of any message flow to the process. That is, the delivery of an alarm to a process cannot be blocked by any messages queued for delivery to the process. Unlike a message which can carry a substantial amount of information, the information conveyed by an alarm is limited to a very short alarm code. This limitation implies that the delivery of alarms can be accomplished in a way that requires little in the way of communication or storage resources. This makes it possible for MSG to insure certain "priority" treatment for alarms which makes them suitable for alerting processes to exceptional events. While similar to traditional interrupts, alarms are different in one important respect: the delivery of an alarm to a process does not necessarily imply that the process is subjected to a forced transfer of control by MSG. For this reason, we have chosen to use the term alarm rather than interrupt.

All modes of interprocess communication supported by MSG follow the same basic pattern, which is roughly as follows:

1. One process tells MSG about a message or alarm to be sent or a connection to be opened. It also specifies a destination address and a signal by which MSG can inform it that the message or alarm has been sent or the connection opened.

2. Another process which matches the above destination address tells MSG that it is ready to receive the same type of communication. It also specifies a signal by which MSG can inform this process that the message or alarm has been received or the connection opened.

3. MSG sends the alarm or message or opens the connection. It also signals the source process that the message or alarm has been sent or the connection opened and signals the destination process that the message or alarm has been delivered or the connection opened. After it receives the signal, the process receiving a message or alarm always knows the specific address of the sending process.

C. The NSW Foreman Component

As part of our participation in the NSW system design effort, this quarter we began to design and document the functional requirements of the NSW Foreman component. The Foreman is the local-to-the-tool component of the NSW system. Each tool has a Foreman which is responsible for the startup and termination of the tool, and the smooth operation of the tool with the other NSW components. In conjunction with the facilities supported by the Works Manager Component, the Foreman provides the tool with its NSW runtime environment. Every tool runs under the control of a Foreman.

The Foreman has two (to be) well defined parallel interfaces. One interface is between the Works Manager Process and the Foreman. This interface is organized around the MSG message passing capability, and involves both the WM instructing the Foreman about handling the tool, and the Foreman requesting WM services on behalf of its tool. The other well defined interface is between the Foreman and its tool. The Foreman maintains an operating system like interface to the tool. It is through this interface that the tool can invoke the various functions that distinguish the NSW environment from the host operating system environment.

After careful consideration, we have identified four aspects of the functioning Foreman which need attention. They are providing for tool startup/initialization/termination, providing the NSW runtime environment for tools written to function in the NSW,

providing for encapsulation of old programs written exclusively for the local host operating system, and providing mechanisms to allow the debugging of tools. The Foreman specification document, scheduled for completion in the next quarter, will detail initial efforts in defining the NSW needs in these cases.

III. TENEX Related Activities

A paper, entitled "The TENEX Pie-Slice Scheduler" was submitted for presentation at the National Computer Conference in June. In the course of writing this paper, a design error in the computation of the scheduling priority function was discovered, which affected the ability of the scheduler to provide pie-slice groups having non-uniform demand with their fair share of the processor. This error has been corrected, with the updated scheduler now running on all four BBN TENEXs.

During this quarter the initial design for the secure TENEX monitor was developed in conjunction with MITRE Corporation. This design was presented to both the Lampson Security Committee and to the Mailsystems designers. The latter meeting revealed several ways in which the security measures being added to TENEX made design of the mail systems difficult. Most of these points were subsequently resolved and it is now felt that the modifications to the TENEX monitor are sufficiently solid that actually coding may begin.

IV. MAILSYS Development

This report covers progress in message technology during November and December of 1975. Subsequent to that time, the message technology effort has continued under a new contract with different reporting cycle. The first report under that contract will cover progress in January, February and March of 1976.

Our major effort during the November-December period has been to move ahead systematically with the development and refinement of MAILSYS/HERMES Version 2. We reported in the last time period on the creation of a phased development plan covering HERMES 2.2 through 2.5. Versions 2.2 and 2.3 were accomplished during the current reporting period, as described in the following pages.

The current phased development effort has been guided in large part by the human factors evaluation of the early, experimental states of HERMES 2. That effort is carried on continuously and has resulted in further refinements, improvements and extensions for present and future versions of the system.

In a related effort, our human factors group has begun development of documentation aimed at providing tutorial support to end users of the system with varying levels of sophistication in computer technology.

Work on the LDMX interface has continued with collaboration between BBN and NAVCOSSACT on the details of the LDMX/TENEX protocols. At this writing the final design for the TENEX portion

of the protocols has been completed and coding is approximately 90 percent complete.

Documentation of HERMES has continued with development of on-line documentation material in parallel with development of the system itself. We have focussed both on the content of the documentation and its structure -- the internal organization and coupled software supporting mechanisms that will deliver the documentation to the user in a structured form.

Monitor-level security work related to message system development has also proceeded during this period. Our work in the security area has been described in Section III above.

In November and December 1975, we released two new versions of MAILSYS/HERMES, each embodying important changes in the the system.

A. MAILSYS/HERMES Version 2.2 "Lower-Level Message Creation"

In MAILSYS 2.1, and in all previous versions of MAILSYS, the commands for creating messages were merged with all other commands at the "top level" of the system (with prompt signal >). Version 2.2 embodies the concept discussed in the last Quarterly Progress report under V.A.3 "Other Extensions".

The message-creating commands have been separated in a subcommand-level unit, equivalent to the units that create, edit and display other objects in the system, such as templates and sequences.

To enter the message-creating mode, the user types either CREATE<CR> or EDIT DRAFT<CR>. MAILSYS responds with a double prompt signal >>, and the message-creating subcommands become available.

The message currently being created is considered to be a coherent object, called a "draft". The user can enter and leave the subcommand mode without disturbing the contents of the current Draft.

Separating out the message-creating commands has allowed us to combine the two commands SHOW <object> and DISPLAY <field> into a single command SHOW. At the top level, SHOW prints out the contents of any object in the system, including the current draft. At the message creating level, SHOW prints the current draft, and SHOW <field> prints the contents of an individual field.

Note that, where possible, we are trying to eliminate excess commands in this manner. We believe that there should be a relatively small number of "verb concepts". Differences in command behavior should be achieved by modifying the behavior of a single verb (through a choice of objects, modifying clauses, etc.) rather than by a proliferation of different verb names.

We have revised the system of modifiers used with SHOW. HERMES can now SHOW ALL objects, or objects in a certain class, such as sequences or templates. In addition, HERMES can SHOW names only or current objects only or complete objects. We have considerably improved the display of SHOW SWITCHES. A human factors analysis of the presentation of switch options led to a tabular display in which the different optional SWITCH settings are always shown in the same relative positions, and the current settings are indicated by distinctive pointers. The user can tell at a glance which settings differ from standard HERMES settings and what other optional settings are available.

We have improved the operation of the SEND command, and made it much faster, by making full use of the PMAPping features of the TENEX Operating System.

We have improved the logic for handling the Control-O character. It now aborts any print-out in the system, such as the Herald, the Initial Survey or the output of SHOW.

The MAILSYS/HERMES overwrite option for files has been changed to conform with the normal file-handling conventions in TENEX, and we have added a default option for the FILE command that makes it possible for multiple file requests to address the same destination.

We have added a protection mechanism to guard against conflicts when two users work with the same message-file. If two users both have permission to add to or delete from a message file, and both try to access the file at the same time, the first user to access the file now takes possession of the "write access" to the file. The second user is able to read messages but not to delete or to file new messages. HERMES warns the second user about the situation.

It is now possible to use the DESCRIBE command to direct on-line documentation to a destination other than the User's terminal, such as the lineprinter or another file.

The LIST command also, can now be directed to a destination other than the lineprinter, such as the User's terminal or another file.

The User's Profile now contains a MESSAGE-FILE RECORD that keeps track of all message files accessed by the User, and the date they were last accessed. This information puts the definition of RECENT on a User basis rather than a file basis. When several Users share a file, each User will be shown the messages that have arrived since the last time he accessed the file, no matter who else may have accessed it in the meantime. The MESSAGE-FILE record information can be printed out with the SHOW command, and individual records can be deleted by the User.

B. MAILSYS/HERMES Version 2.3 "Global Name Space"

Ted Myer and Austin Henderson attended a meeting of the Message Service Committee at the University of Southern California, Information Sciences Institute, December 4-5, 1975. At the meeting, they participated in the drafting and adopting by vote of a set of primitive functions and preferred command which are designed to be a common language core for all message systems.

We agreed to make these functions a part of the MAILSYS/HERMES system, and they have been implemented in MAILSYS/HERMES Version 2.3.

We have put considerable effort into integrating the suggestions of the Message Service Committee into MAILSYS/HERMES design, especially in the choice of default values for the sequences acted upon by the message-handling commands agreed upon by the committee: PRINT, NEXT, SUMMARIZE, REPLY, FORWARD and DELETE.

We have simplified the command structure of MAILSYS by considering all object names as belonging to a single "global" pool. Commands like EDIT and COPY no longer require the user to specify the type of object to be acted upon, but instead require only the name.

Ex.: COPY JOE instead of COPY TEMPLATE JOE

We continue to improve the choice of names for objects and commands. The commands for directly manipulating user-created

objects, including the Draft message, are now CREATE and EDIT. FORGET and ERASE have been combined in ERASE. REMEMBER and COPY have been combined in COPY.

We have improved the template mechanism by including two new template items: "source" and "separate". The "Source" item prints the FROM field of the message unless the FROM field is the same as the User's LOGIN directory; in that case, "Source" prints the first name in the TO field. This is convenient for people who like to keep copies of their messages by sending CC copies to themselves. The MAILSYS-supplied BASIC-SURVEY template now has a "Source" item in place of the "To" item.

The "Separate" item allows the User to specify page separation so that, for example, each message Printed on the lineprinter will start at the top of a new page.

The display output of SHOW has been improved for sequences, also, so that continuous ranges of message-nos. are shown in the same form as the type-in.

Ex.: 4:8 rather than 4 5 6 7 8

This greatly condenses the print-out of the SHOW Sequences command when message-files contain a large number of messages.

A SUGGESTION command has been included that makes it easy to send messages that refer automatically to the particular version of MAILSYS in use.

On an experimental basis, two new commands have been added. EXIT duplicates QUIT, and MOVE duplicates FILE, but each pair has Switches that allow the two members of the pair to function in a slightly different fashion. The EXIT and QUIT (as well as INPUT) switches allow the User to choose between automatic expunging of the message-file, no expunging, or having MAILSYS query the User each time the command is used. MOVE and FILE have the same kind of three-way switch for whether messages should be deleted after they are copied into another message file. The REPLY command has a three-way switch for whether copies should be sent to all names listed in the TO and CC fields.

We are now using the new and improved BCPL compiler which gives faster code, smaller memory required for the code, and a faster compile time.

C. Human Factors

The human factors group has reviewed the selection of names and format for MAILSYS/HERMES commands and objects. The design choices have been made with the intent of producing a logical system that uses clearly defined rules and operates as much as possible in analogous ways in its different parts.

We have begun a study of the effect of the use of CRT graphic display terminals upon the human-machine interface. Our goal is to develop a system that is comfortably compatible with both CRT's and printing terminals.

1. The Human Factors group has continued consulting on the design of the human interface, and has participated in MAILSYS design meetings.

2. A draft document has been prepared which is an introduction to a minimal system, aimed at the COTCO user. This document takes the perspective of a user who will work with a fixed set of system objects that are given to him, and who wants to begin using the system to prepare and read messages.

   This User's Manual does not include references to "filters", "templates", "streams", to the concept of a current environment, or to the User's Profile. It is assumed that each of these will be predefined and configured for a given class of user and that they will be alterable within a particular environment only by authorized personnel.

3. We have continued discussions with MITRE staff members Jonathan Mitchell and Nancy Goodwin about plans for the COTCO interactive test, with particular attention to the effect of CRT graphic display terminals.

D. LDMX Interface

We are continuing our collaboration with NAVCOSSACT to design the details of the protocols of the LDMX/TENEX interface.

1. On 20 November 1975, Frank Ulmer attended a meeting with NAVCOSSACT to discuss the details of the protocols.

   The continuing discussion between NAVCOSSACT and BBN culminated with the approval of changes in the specifications for the Header Control Block format. Some of the restrictions on the relations between the roles of LDMX and TENEX were relaxed, so that now either computer can send most of the block types, although a few block types are uniquely reserved to one computer.

2. The final design for the TENEX portion of the protocols has been completed.

   The design has been simplified and clarified. The translator facility has been removed. In its place, we will use the MAILSYS/HERMES template facility; it will translate from the Military message format to a standard MAILSYS/HERMES format, which can then be modified by the user, if desired.

3. The coding for the TENEX portion of the protocols has been 90 per cent completed. The implementation of the final design was accomplished without requiring further changes in the design.

E. Documentation

In October 1975, we produced two interim pieces of documentation for MAILSYS Version 2. "MAILSYS SUMMARY" and "A CONCEPTUAL INTRODUCTION TO MAILSYS MESSAGE-PROCESSING COMMANDS". These were made available as TENEX files, not integrated into the MAILSYS program, and were updated in November and December 1975 to accommodate changes in MAILSYS Version 2.

We began work on a general rewriting of the material generated by the DESCRIBE command, and on a new reference document for the experienced user that lists all MAILSYS commands in capsule form.

Work was also begun on the design of system documentation which will integrate the descriptions of individual commands and objects in the system with a general over-all description and with examples of the use of HERMES.

The documentation will be accessible at many levels, possibly through a new, structured documentation command, and the user will be able to direct the output to a terminal, a lineprinter, or a TENEX job.

As an interim measure, and as an experiment in using MAILSYS facilities, MAILSYS-readable documentation was released for MAILSYS Version 1, which is currently in active use at BBN on all four TENEX systems. A TENEX file has been set up that contains a series of documentation messages. The users can access the message-file through MAILSYS and use MAILSYS to print the messages on either

terminals or the lineprinter. The messages contain information that was formerly available only when it was printed on the User's terminal with the HELP, NEWS and DESCRIBE commands. The messages were edited to eliminate the duplication inherent in the DESCRIBE material and to add some reference tables.